PAL Consolidation 15

University of Hertfordshire
– Week 17 – March 2016 –

Aim & Objectives

- Discuss previous session
- Look forward to next session

Introduction

In this session we will look at reading and writing to Java and making a Java GUI. The program to demonstrate all of these things will be a simple text editor.

Exercise - Saving and reading files in Java

So for this, we will make a few assumptions, as follows:

- The files that need to be read and written are in text.
- The files are small enough to fit into '`RAM`' and the '`JVM`'.
- The files don't overflow the size of a single '`String`' object.

Using those assumptions, we can use the following code to read a file into a '`String`':

```
/**
   * readFile()
   *
   * Read the files into a String object.
   *
   * @param file The file to be read.
   * @return The contents on the file,
otherwise NULL.
   **/
private String readFile(File file){
   String data = null;
   FileReader fr = null;
   try{
      fr = new FileReader(file);
      /* Create a buffer to read the bytes
into */
      char[] buffer = new
char[(int)file.length()];
      /* Make the read */
      fr.read(buffer);
      data = new String(buffer);
      /* Close the file for everybody else */
      fr.close();
   }catch(IOException e){
      /* Do nothing */
   }finally{
      /* If we did manage to open the file,
don't lock it up */
      if(fr != null){
         /* Try to finish what we started */
         try{
            fr.close();
         }catch(IOException e){
            /* Do nothing */
         }
      }
   }
   return data;
}
```

And the following will write a file:

```
/**
   * writeFile()
   *
   * Writes the file to the disk.
   *
   * @param file The file to be written.
   * @param data The data be be written to
the disk.
   **/
private void writeFile(File file, String
data){
   FileOutputStream out = null;
   try{
      out = new FileOutputStream(file);
   }catch(FileNotFoundException e){
      /* Handle error */
      writeError();
   }
   try{
      if(out != null){
         out.write(data.getBytes());
         out.close();
      }
   }catch(IOException e){
      /* Handle error */
      writeError();
   }
}
```

Exercise - GUI

To implement the GUI, you'll need to look at the following concepts in Java:

- '`JFrame`' - This will be for the main window.
- '`JTextArea`' - This will be for the text in your window.

There are many guides to creating a GUI, using the read and write code given to you produce a basic text editor. An example program will be given at the end of the session.

Exercise - Further Improvements

Here we should start to look at fixing the above limitations. We can also start to do some other interesting ideas as follows:

- **Syntax highlighting** - in particular, look at the syntax highlighter written at `http://homepages.herts.ac.uk/~db12aba/code-highlight.js` for a method of highlighting without knowing the language.
- **File types** - How should we be reading and writing different files in ASCII, Unicode, etc?
- **Display characters** - Unicode has a tonne on emoji and other useful characters - is there some way we can begin to start displaying those?
- **File tracking** - Detecting when a file we're working on changes. Java's '`File`' class offers some nice information on this front.

Resources & Further Reading

'*http://homepages.herts.ac.uk/~db12aba/*' – All content from these sessions updated weekly.

'*http://code.org/*' – A good resource testing your programming skills.

'*http://stackoverflow.com/*' – Highly recommended online help for programmers (NOTE: Employers are interested to know whether you're an active member of this site!).

‘*http://draw.io*’ – A very good, free online drawing tool that exports to many formats, including ‘XML’ and ‘JPG’.